Efficient Availability Assessment of Reconfigurable Complex Multi-State Systems with Interdependencies

Hindolo George-Williams
Institute for Risk & Uncertainty Engineering
University of Liverpool, UK

Institute of Nuclear Engineering & Science
National Tsing-Hua University, Taiwan

Edoardo Patelli
Institute for Risk & Uncertainty Engineering
University of Liverpool, UK

ABSTRACT: Complex topology, multi-state behaviour, component interdependencies and interactions with external phenomena are prominent attributes of many realistic systems. Analytical reliability evaluation techniques have limited applicability to such systems, and efficient simulation models are therefore required. In this paper, we present a simulation framework to simplify the availability assessment of these systems. It allows the tracking of changes in the performance levels of components, from which system performance is deduced by solving a set of flow equations. This framework is adapted to the availability modelling of an offshore plant with interdependencies, operated in the presence of limited maintenance teams and operational loops. The result is compared to a Monte Carlo simulation-based solution in literature that required the enumeration of the plant's *cut sets*. The proposed approach is shown to be more intuitive, robust to errors and require less human effort.

1 INTRODUCTION

Most of the systems we interact with on a day-to-day basis exist as interdependent systems. Two systems are interdependent if at least two nodes (one from each system) are coupled by some phenomena such that a malfunction of one affects the other. The coupling phenomenon could be proximity in space/geographic locations (Buldyrev, Parshani, Paul, Stanley, & Havlin 2010) or functional interdependence (Zio & Sansavini 2011). A water distribution network, where pumps and other electrical driven appliances rely on the reliability of the power grid is a typical example.

System components are normally prone to random failures arising from their intrinsic properties or induced failures from targeted attacks. In interdependent systems, an undesirable glitch in one system could cascade and cause catastrophic disruptions in coupled systems. The cascade could be fed back into the initiating system and the overall consequences may be unimaginable (Buldyrev, Parshani, Paul, Stanley, & Havlin 2010). This was made clear by the massive blackout that struck Italy in September 2003, affecting the internet network in the process. To minimize the effects of cascading failures, some interdependent systems are equipped with reconfiguration provisions. This normally entails transferring operation to another node, rerouteing flow through alternative paths or shutting down parts of the system.

Various models have been developed to study the effects of interdependencies on systems. However, a good number of these only assess their response to targeted attacks, variation in some coupling factor or the relative importance of system nodes (Rosato, Issacharoff, Tiriticco, Meloni, Porcellinis, & Setola 2008, Buldyrev, Parshani, Paul, Stanley, & Havlin 2010, Zio & Sansavini 2011). When faced with the situation of random node failures, a complete reliability and availability analysis should be performed. Renowned analytical multi-state system reliability evaluation techniques, like Binary Decision Diagrams (Zang, Wang, Sun, & Trivedi 2003), Sum-of-Disjoint-Products (Yeh 2015), and the Universal Generating Function (Levitin 2005), however, are of very little use to the evaluation of these systems. Their inapplicability is amplified if, nodes can undergo non-Markovian transitions, their restoration can be de-

layed or the system is reconfigurable. Such considerations are only implementable by simulation algorithms (Zio, Baraldi, & Patelli 2006).

However, most multi-state system simulation algorithms rely either on the structure function of the system or enumeration of the system's path or cut sets (Zio, Baraldi, & Patelli 2006, Ramirez-Marquez & Coit 2005). Both procedures get cumbersome even for complex systems of moderate size, and with them, shut down and restart of components (a type of reconfiguration) is non-intuitive (George-Williams & Patelli 2016).

In this work, a novel simulation approach is presented to overcome these challenges. The approach defines the system as a directed graph and uses the interior-point algorithm (Mehrotra 1992) to determine the magnitude of flow along each edge of the graph for a given performance level of its components. This enhances system simulation without the need for its state enumeration and subsequent path/cut set definition. Since the actual flow through every node can be calculated, shut down and restart of nodes is more intuitive, regardless of system architecture. Using the system originally presented by (Zio, Baraldi, & Patelli 2006), we illustrate the application of this approach to interdependent systems and systems prone to limited maintenance teams.

The remainder of this paper is organised as follows; Section 2 is dedicated to formulating the problem under consideration. In Section 3, the system reliability approach is described, and the implementation of the problem is provided in Section 4. Section 5 discusses the proposed approach with respect to its computational efficiency and other dynamics. A conclusion to the work is contained in Section 6.

2 PROBLEM FORMULATION

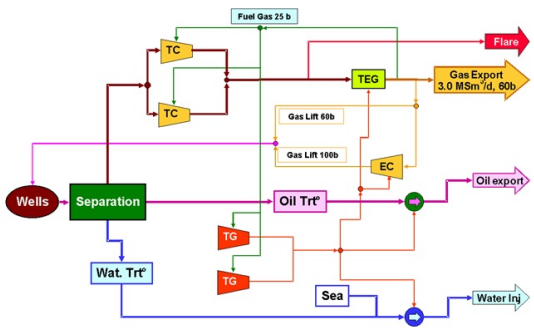
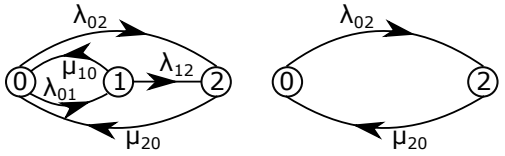


Figure 1: Schematic of offshore installation

The following notations are used, **TG**, Turbo Generator; **TC**, Turbo Compressor; **TEG**, Try-ethylene Glycol Unit; **EC**, Electro-Compressor; λ_{mn}, failure rate from state m to n; μ_{mn}, repair rate from state m to n.

Figure 1 shows the schematic of an offshore installation and Figure 2, a description of the failure



(a) TCs and TGs (b) TEG and EC

Figure 2: State-space diagrams of components

and repair transitions of six of its components, the remainder are assumed perfectly reliable. State 0 represents a component in its normal operating mode and state 1, its partial failure. When partially failed, the component maintains its nominal performance but with an increased failure probability to state 2, where it is completely failed. The Well nominally produces 5.0×10^6 units of gas, 26500 units of oil and 8000 units of water a day. These are separated at the separation unit and transmitted via independent dedicated paths as shown in Figure 1. The nominal gas demand is 3×10^6 units at 60bar and when gas production exceeds demand, for safety reasons, the excess is burnt as flare. Additional details of the offshore plant are available in (Zio, Baraldi, & Patelli 2006).

2.1 *Interdependencies & Reconfiguration*

The major components of the plant (TEG, EC, oil pump and water pump) require continuous supply of electricity to function. This reliance creates a functional coupling between the electricity network and the paths transporting the three commodities. A second functional coupling is introduced by the reliance of TCs and TGs on dried compressed gas for their functioning. Each TG is rated 13MW, the TEG and EC consume 6MW each while the two pumps consume 7MW each. When only one TG is available, the EC and the water pump are shut down to maintain the production of oil and gas. To ensure nominal production, a fraction of dried compressed gas (1.0×10^6 units) is diverted, compressed in the EC to 100bar and re-injected into the well. If the EC is unavailable, the gas is directly injected into the well at 60bar resulting in a production at 80% of nominal levels. With no gas injection, production level drops to 60%.

2.2 *Maintenance Policy*

Corrective maintenance (CM) is carried out according to a predefined priority. Repairs once initiated remain unaffected by failure of other components, regardless of their superiority on the priority list. In addition to CM, TCs and TGs undergo three types of preventive maintenance (PM) interventions whilst the EC undergoes one. To ensure minimal effect on performance, PM is carried out only when the system is perfect. PM interval is defined as the absolute time between

successive PM interventions.

2.3 *Monte Carlo Simulation*

In (Zio, Baraldi, & Patelli 2006), the goal was to determine the production availability of the plant under the maintenance policy described. It was approached by enumerating the plant's production levels, reconstructing the cycle of component failures & maintenance and monitoring production level occurrences. Identifying the production level corresponding to a given plant configuration during the simulation had required the use of an innovative approach based on cut sets. In practice, each production level is identified by a pair of cut sets defined as minimum and maximum cut sets. Although the solution proposed was very efficient and innovative, it required the manual identification of those cut sets and production levels. This, even for a moderately sized system can be quite time consuming, impossible for complex system structures, error prone and involves considerable human effort.

3 PROPOSED APPROACH

Given the challenges of the Monte Carlo Simulation described in Section 2.3, a simple, intuitive and generally applicable simulation approach is developed and applied to the system described in Section 2. Like in (Zio, Baraldi, & Patelli 2006), we mimic the operation of the system by studying the cycle of component transitions. Instead of manually listing possible system performance levels and cut-sets, system performance is deduced by a linear programming algorithm. For each transition resulting in a new system performance, both the new performance and its time of occurrence are recorded. These recorded histories are used at the end of the simulation to determine the relevant reliability and performance indices by the procedure described in (Zio, Baraldi, & Patelli 2006).

In this section, the relevant mathematical principles are derived and the simulation model described. These underlying principles are based on an extension of the load-flow simulation model recently presented by the current authors. Therefore, only the important modifications are presented, details are available in (George-Williams & Patelli 2016).

3.1 *The Component Model*

In (George-Williams & Patelli 2016), a component is defined by a quintuple that takes into account various properties of the component. Properties like load capacity, flow losses and minimum threshold load, excluding component interdependencies are considered. In order to adapt the model to components with interdependencies, two additional parameters; $\mathbf{L}$ and $\mathbf{D}$ are introduced. $\mathbf{L}_i = \{k, l\}$ defines node i's load dependency on node k. Load dependency here means node i requires a minimum of l level of flow from node k to operate. Under this condition, a load-source relationship is said to exist between the nodes which may belong to different subsystems. $\mathbf{D}_i = \{d_{j1}, d_{j2}, d_{j3}, d_{j4}\}_{u\times 4} \mid j = 1, 2, ..., u-1, u$ defines the single-way causal-effect relationship between node i and other nodes. This type of coupling specifies induced state changes in other nodes following a state change in i. d_{j1} is the state of i triggering the event, d_{j2}, the affected node, d_{j3}, the state the node has to be in to be affected and d_{j4}, its target state on occurrence of the event. Each row of $\mathbf{D}_i$, therefore, defines the behaviour of an affected node and u is the number of relationships.

3.2 *The System Model*

To enable the determination of a system's performance without reference to its cut-sets, its structure is defined by a directed graph which nodes represent system components and edges; their physical links (George-Williams & Patelli 2016). If these components, including demand points are numbered consecutively from 1 to M, then, the system can be defined by an adjacency matrix, $\mathbf{A}$. $\mathbf{A}$ is such that each link between connected components is represented by 1 in the relevant position as defined in Equation 1.

$$\mathbf{A} = \{a_{ij}\}_{M\times M} \mid a_{ij} = \begin{cases} 1 & \text{If flow is } i \rightarrow j \\ 0 & \text{Otherwise} \end{cases} \tag{1}$$

The edges of the graph are defined by a k by 2 matrix; $\mathbf{e}$, where k, the number of edges is equal to the total number of non-zero elements in $\mathbf{A}$. Edge e_{ij} depicts the edge originating from node i and terminating on node j. $\mathbf{e}$ is obtained by traversing $\mathbf{A}$ from the upper left to the lower right element, exploring each column from top to bottom and extracting i and j for each non-zero entry. Its properties are outlined in Equation 2, where $\mathbf{V} = \{1, 2, ..., M\}$ is the set specifying the nodes of the system. The graph can be defined by a single parameter $G \mid G = (\mathbf{V}, \mathbf{A})$.

$$\mathbf{e} = \{i, j\}_{k\times 2} \mid k = \sum_{j=1}^{M}\sum_{i=1}^{M} a_{ij} \quad \forall (i,j) \in \mathbf{V} \tag{2}$$

The incidence matrix; $\mathbf{\Gamma}$ defines the relationship between nodes and edges and it's related to $\mathbf{A}$ by Equation 3. The variable $q = 1, 2, ..., k$ (the edge number) is the index of edge e_{ij} in $\mathbf{e}$ and $p = 1, 2, ..., M$.

$$\mathbf{\Gamma} = \{\gamma_{pq}\}_{M\times k} \mid \gamma_{pq} = \begin{cases} 1, & p = i \\ -a_{ij}, & p = j \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$\forall (i,j) \in \mathbf{e}$$

$\mathbf{\Gamma}$ is obtained by looping over the rows of $\mathbf{e}$ and updating the former according to Equation 3. George-Williams et al (George-Williams & Patelli 2016) devised an algorithm to compute $\mathbf{\Gamma}$ and $\mathbf{e}$.

3.2.1 *System Flow Equations*

With the adjacency matrix defined and $\mathbf{\Gamma}$ derived, derivation of the system's flow equations is the next important task. These equations take into consideration various node and link performance parameters and are obtained by applying the principles of flow conservation across every system node. Two sets of linear equations which coefficient matrices have an established relationship with $\mathbf{\Gamma}$ are obtained. Therefore, by only defining the adjacency matrix; $\mathbf{A}$, $\mathbf{\Gamma}$ and subsequently the coefficient matrices can be easily obtained. The details of this procedure are outside the scope of this paper but sufficient information is available in (George-Williams & Patelli 2016).

3.2.2 *System Flow Calculation*

Simulation normally entails repeated calculation of system output as nodes undergo their transition cycles. Calling the interior-point algorithm for every transition may impose unprecedented computational burden. This is because a certain system configuration may be attained more than once, making multiple calculations for the same configuration a possibility. To mitigate this, it's desirable to determine node flows for all possible combinations of system node performance levels prior to simulation. Let $\boldsymbol{\beta}$ be the matrix holding these combinations and $\mathbf{C}_u^{\{i\}}$, the set of unique performance levels of node i. $\boldsymbol{\beta}$ is an $M \times \prod_i^M n_i$ matrix; M being the total number of nodes and n_i, the number of unique performance levels of node i. For instance, if the performance of node 3 is defined by $\mathbf{C} = \{10, 0, 0, 10\}$, $\mathbf{C}_u^{\{3\}} = \{0, 10\}$ and $n_3 = 2$. For each combination of performance levels, the corresponding node flows are calculated and recorded. During simulation, $\boldsymbol{\beta}$ is searched for the combination of node performances corresponding to the current system configuration and its pre-stored node flows simply read off. By this, flow calculation for every configuration is carried out only once.

3.3 *Node Transition Parameters*

Determining the transition time, t_{next} of nodes is the core of every simulation algorithm. Given the current state, x, of a node, all possible transitions from x are sampled and their minimum value, t_{min}, selected. The transition producing t_{min} is the node's next transition, occurring at $t_{next} = t + t_{min}$, t being the current simulation time. If t_{min} is associated with multiple transitions, one of them is randomly selected as specified by the sampling algorithm in (George-Williams & Patelli 2016).

3.4 *Accounting for Dependencies*

Let $\boldsymbol{\mu}$ be the vector holding the current performance levels/capacities of system nodes. When node i makes a transition resulting in a change in its performance level, the current capacity of its load-source pair, k, (see Section 3.1) is modified. If $c_{x^-}^{\{i\}}$ is the node's capacity before transition, and $c_x^{\{i\}}$, its current capacity, then the change in the capacity of node k is defined by,

$$\boldsymbol{\mu}(k) = \begin{cases} 0 & \text{If } c_{x^-}^{\{i\}} > 0 \text{ and } c_x^{\{i\}} = 0 \\ l & \text{If } c_{x^-}^{\{i\}} = 0 \text{ and } c_x^{\{i\}} > 0 \end{cases} \tag{4}$$

A recursive algorithm is required to account for the causal-effect relationships between nodes, owing to the possibility of nested dependencies. If $\mathbf{D}_i$ and x_i are respectively the dependency matrix and current state of node i, then, the algorithm is summarized thus;

1. Find all nodes affected by the state change (using $\mathbf{D}_i$ and x_i) and save in a register, their next states.
2. Select the last entry, node j, of the register, force the transition specified by $\mathbf{D}_i$ and delete its records.
3. Using $\mathbf{D}_j$ and x_j obtained in step 2, repeat steps 1 and 2.
4. Repeat steps 1 through 3 until the register is empty.

On each node transition, $\boldsymbol{\mu}$ is updated and any load dependencies accounted for as earlier described.

3.5 *Forcing Maintenance*

Algorithm 1 Forcing maintenance with limited teams

Require: m_1, m_1', m_2, m_2', h_1 and h_2

```
k ← 1                                   ▷ initialize indicator
while k ≤ 2 do
    v ← m_k − m'_k                      ▷ get idle teams
    while v > 0 and h_k ≠ ∅ do
        select node according to priority
        make maintenance state its current state x
        sample its next transition using x
        delete node from h_k
        v ← v − 1,   m'_k = m'_k + 1
    end while
    k ← k + 1
end while
```

With limited maintenance teams, maintenance actions are not instantaneous. Therefore, the transition from a degraded state or to PM has to be manually executed during simulations. Let m_1 & m_2 respectively denote the number of teams dedicated to CM and PM and m_1' & m_2', the number of busy CM and PM teams at time t. Following a transition, a component is added to the set h_1 of components requiring repairs if the new state is directly linked to a CM state and to h_2 if

its PM is due. Algorithm 1 describes the procedure to force maintenance. If PM is carried out only when the system is perfect, the algorithm is terminated after the task for $k = 1$ if at least one of $h_1 \neq \emptyset$, $m'_1 > 0$ and $m'_2 > 0$ is true. Each of these conditions means either there is a failed component waiting to be repaired or maintenance is in progress; any of which suggests the system is not in a perfect state.

3.6 *Node Shutdown and Restart*

Node i is shut down if its flow falls to or below its threshold, Λ_i, or if flow through its load-source pair, k, falls below l. If a node is shut down, its current and next states are saved, its next transition time set to ∞ and its current capacity, to 0. When the condition leading to shut down is resolved, the node is restarted and restored to its previous state. The period, t_{shift}, spent in shut down is accounted for by shifting its next failure time to $t'_{next} + t_{shift}$, where t'_{next} is its failure time before shut down.

4 IMPLEMENTATION

Here, a chronological description of the application of the principles and algorithms described in Section 3 to the system presented in Section 2 is given. Figures 3 and 4 are modifications of the state diagrams of the plant's components presented in Figure 2. The modifications are such that the realistic operation of components, consequent of system dynamics is reflected. The state, *Shutdown*, is introduced to account for restart and shut down (reconfiguration of the component). In the plant, PM takes place only when all its components are in their perfect states. This explains why the transition to PM in the modified state diagrams is from state 0. With the exception of state 4 in Figure 4, a component has 0 capacity when in any of the Shutdown, CM and PM states. State 4 is an exception since the transition represents an on-line repair and there is no need to take the component out of operation. Hence, its capacity from state 2 is retained. Transitions to the maintenance states; CM and PM are forced, as they depend on the availability of an idle maintenance team. For instance, a component remains in state 3 indefinitely until an idle maintenance team commences its repair. Similarly, transitions to *Shutdown* are forced, since they represent induced unavailability of a component due to the unavailability of another component.

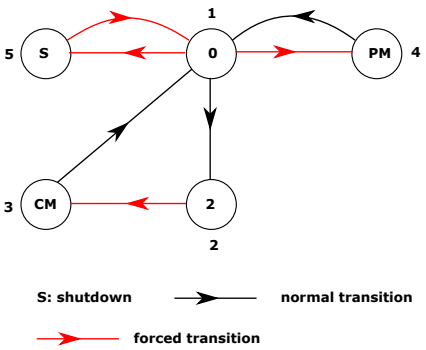


Figure 3: Modified state-space diagram for EC and TEG

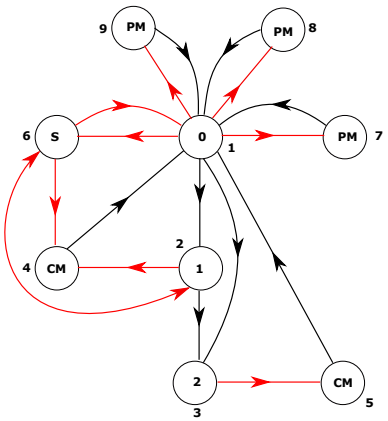


Figure 4: Modified state-space diagram for TCs and TGs

4.1 *Modelling the Plant*

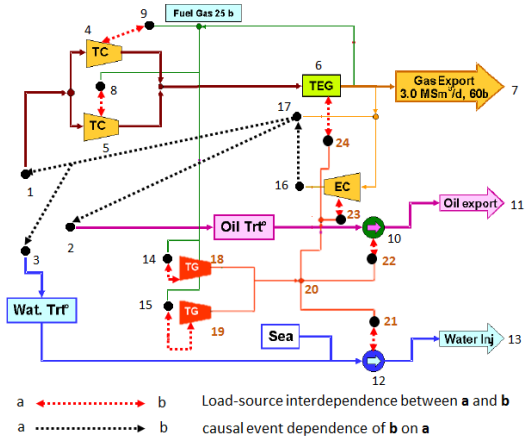


Figure 5: System model showing dependencies

Shown in Figure 5 is the plant's schematic, with the relevant nodes and their relationships. The Well is separated into three nodes, 1, 2, and 3, each supplying gas, oil and water respectively. Its third production level is unreachable, since there is no gas lift only if the entire system is failed. Therefore, each of the three nodes exists in only two states; 100% (state 1) and 80% (state 2) nominal output. A third state, with capacity 0 is introduced to account for the period when the plant is completely shut down, consequent of either PM or component failure. Transitions between the non-zero output levels is triggered by the EC (node 16) and are therefore considered forced transitions. The alternative path for gas lift; node 17, is activated on unavailability of node 16 and deactivated when available. It, therefore, has a standby relationship with the latter and exists in two states; *active* (state 1) and *standby* (state 2). Nodes 1, 2, and 3 are affected accordingly on its activation or deactivation, as specified in Equation 5.

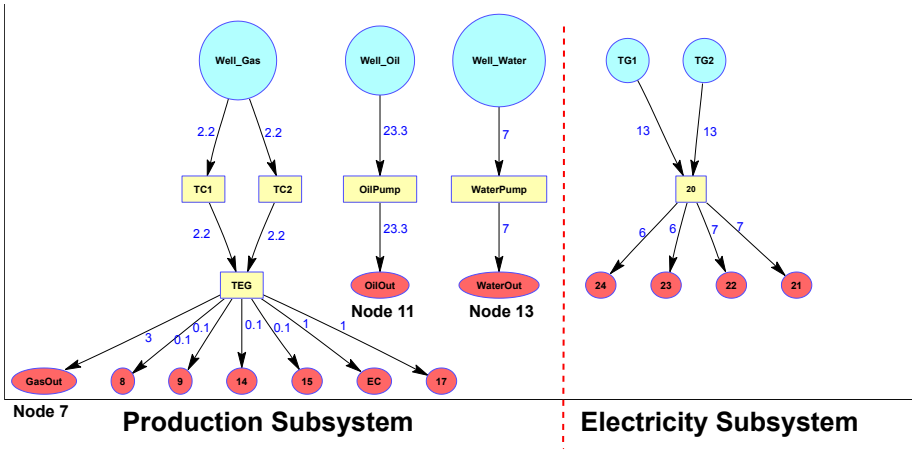


Figure 6: Plant network model

$$\mathbf{D}_{16} = \begin{pmatrix} 1 & 17 & 1 & 2 \\ 2 & 17 & 2 & 1 \\ 5 & 17 & 2 & 1 \end{pmatrix}, \quad \mathbf{D}_{17} = \begin{pmatrix} 1 & 1 & 1 & 2 \\ 1 & 2 & 1 & 2 \\ 1 & 3 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 2 & 3 & 2 & 1 \end{pmatrix} \tag{5}$$

The plant is separated into two subsystems, on the basis of commodity transported. The paths transporting gas, oil and water are considered a single subsystem (production subsystem) by virtue of their independence. The flare is excess gas and it's therefore discarded, since it has no effect on the gas output. Demands at nodes 7, 11 and 13 are respectively taken as the nominal gas output and maximum well output of oil and water. The capacities of nodes 8, 9, 14 and 15 are each 0.1×10^6 units of gas and those of nodes 16 and 17; 1×10^6 units of gas. These nodes, according to the plant's schematic (Figure 5) appear to be competing with node 7 for the gas output from the TEG. In reality, the quantity of gas required to keep the the TGs and TCs in operation and the gas lift are used up first and any excess exported via node 7. However, this is not considered by the subsystem's network model. Therefore, the gas output (flow through node 7), as deduced from the network model should be corrected. The effective gas output is given by the difference between the flow through the TEG and the sum of the quantities used for gas lift ($X_{lift} = 1 \times 10^6$) and by any available TCs and TGs. Following flow calculation, the outputs; X_{gas}, X_{oil} and X_{water} of the three commodities are given by,

$$X_{gas} = \mathbf{s}\left(\eta_6 - X_{lift} - \sum_{i \in \mathbf{w}} c_x^{\{i\}}\right) \tag{6}$$

$$X_{oil} = \eta_{11}, \qquad X_{water} = \eta_{13}$$

Where, η_i is flow through node i, $\mathbf{w} = \{8, 9, 14, 15\}$ and $\mathbf{s}$ is an indicator function that takes the value 1 when $\eta_7 > 0$ and 0, otherwise. Shown in Figure 6 is the plant's network model with the maximum flow along each link indicated. Flow along the gas production line is in Mega units, the oil and water lines; in kilo units and the electricity line; in MW.

The electricity network is considered another subsystem, as shown in Figure 6. Nodes 21 to 24 are demand points (local sources) for nodes 12, 10, 16 and 6 from the production subsystem. They, therefore, exist in two states, *active*, when their respective dependent nodes are working and *inactive*, otherwise. Node 20 is a dummy node, assumed perfectly reliable and assigned a constant capacity of 26 units; the combined maximum output of the TGs. The minimum threshold flows; Λ_{21} and Λ_{23} of nodes 21 and 23 are set to 5.99 and 6.99 units respectively to account for the unavailability of one TG. With only one TG available, flow through nodes 21 and 23 fall below their threshold, and are shut down as outlined in Section 3.6. This augments flows through nodes 22 and 24 to their required levels and keeps the EC and the oil pump in operation. Demand at the three output nodes is fixed, the pumps and node 20 are perfectly reliable. Their reconfiguration (shut down and restart), therefore, is unnecessary. This condition has been represented by assigning a negative value to their threshold flows. With this, the shut down requirement due to their effective load is never satisfied since actual node flows are non-negative. The remaining nodes are assigned a 0 minimum threshold flow.

4.2 *Production Level Determination*

To determine the production availability of the plant, the evolution of X_{gas}, X_{oil} and X_{water} are recorded, as the simulation progresses. At the end, the possible performance levels of each commodity are determined from its performance history. The possible combinations of performance levels of the three commodities are generated and their occurrences in the simulation history identified, to deduce the possible plant performance levels. Owing to interdependencies, a state change in one node may give rise to state changes in a series of other nodes. The system may go

through a number of production levels in the process but the effective production level is the one attained after the last node state change. A recursive algorithm is employed for this purpose as outlined thus,

1. Select the subsystem affected by the last node transition and calculate its flow. If both subsystems are affected, select the production subsystem. Go to step 4 if no nodes are affected.

2. Restart and shut down nodes where applicable. When any of nodes 21-24 and nodes in **w** is shut down or restarted, its load-source pair is also shut down or restarted.

3. Repeat steps 1 and 2, making sure interdependencies are accounted for and $\boldsymbol{\mu}$ updated on every transition.

4. Calculate X_{gas}, X_{oil}, X_{water} and exit algorithm.

4.3 *The Simulation Algorithm*

The simulation proceeds by going through component transitions, determining the plant's production level at each transition and saving these as a function of time. The relevant availability and performance indices are derived from the production history. The simulation procedure is summarised thus;

1. Initialise register to store production level history and calculate flows across both subsystems as described in Section 3.2.2. Set mission time t_m and number of simulation samples, N.

2. Set simulation time, $t = 0$ and $\boldsymbol{\mu}_{old} = 0$, where $\boldsymbol{\mu}_{old}$ is a temporary variable. Define initial $\boldsymbol{\mu}$, initial production production levels, sample next transition of nodes and update $\boldsymbol{\tau}$; the register holding their next transition times. Also determine their PM due times, save and Go to step 6.

3. Determine components with transition times equal to t, update their current state, sample their next transition, update $\boldsymbol{\mu}$, $\boldsymbol{\tau}$, h_1 and h_2 where applicable. If a node is just from PM, determine its next PM due time.

4. Force maintenance if there are available teams.

5. Determine production if $\boldsymbol{\mu}_{old}$ and $\boldsymbol{\mu}$ are different.

6. Set $\boldsymbol{\mu}_{old} = \boldsymbol{\mu}$ and $t = min(min(\boldsymbol{\tau}), t_{pm})$, t_{pm} being the time the next node is due for PM. Repeat 3 through 6 until t exceeds t_m.

7. Repeat steps 2 through 6, N times.

8. Get the possible plant production levels and determine the relevant performance indices.

4.4 *Simulation Results*

A matlab application was developed to model the plant under the following scenarios; CM only by one team, CM only by two teams and both CM and PM by two teams; one dedicated to each maintenance type.

Table 1: Production levels identified by simulation algorithm

Output Type	**Production Level**						
	1	**2**	**3**	**4**	**5**	**6**	**7**
Gas ($\times 10^6$)	3	0.9	2.7	1	2.6	0.9	0
Oil ($\times 10^3$)	23.3	23.3	21.2	21.2	21.2	21.2	0
Water ($\times 10^3$)	7	7	0	0	6.4	6.4	0

Table 2: Comparison of production level probabilities

Production Level	**State Probability**		
	Case 1	**Case 2**	**Case 3**
1	9.22×10^{-1}	9.30×10^{-1}	7.74×10^{-1}
2	2.99×10^{-2}	2.74×10^{-2}	8.03×10^{-2}
3	3.84×10^{-2}	3.60×10^{-2}	8.93×10^{-2}
4	2.50×10^{-3}	1.10×10^{-3}	5.90×10^{-3}
5	4.70×10^{-3}	4.70×10^{-3}	4.09×10^{-2}
6	3.11×10^{-4}	1.43×10^{-4}	1.70×10^{-3}
7	1.90×10^{-3}	7.84×10^{-4}	3.80×10^{-3}

Using 10^5 simulation samples and $t_m = 1000$ hours in cases 1 and 2, 3×10^4 samples and $t_m = 2 \times 10^5$ hours in case 3, 7 production levels were identified, as presented in Table 1. Their probabilities of occurrence are presented in Table 2, and Figure 7 shows the instantaneous production levels with no PM. As

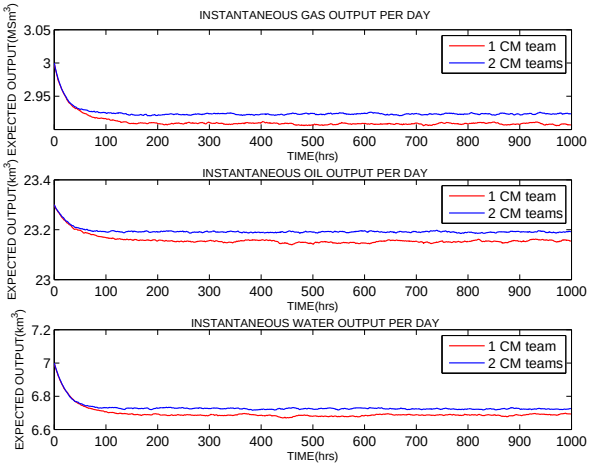


Figure 7: Instantaneous plant performance under CM only

expected, the plant performs better with two maintenance teams. Overall, its availability at the nominal level improves, albeit slightly (see Table 2). However, both scenarios yield the same performance within the first 30 to 45 hours of operation. This is explained by the high initial reliability of components, such that there are only a few failures which can be handled by a single team. As fatigue creeps in, failed components begin to queue and more than one maintenance team is required. It's also evident in Table 2 that the overall performance drops with PM. This behaviour is attributed to the fact that components exhibit exponential failure characteristics, PM only increases their

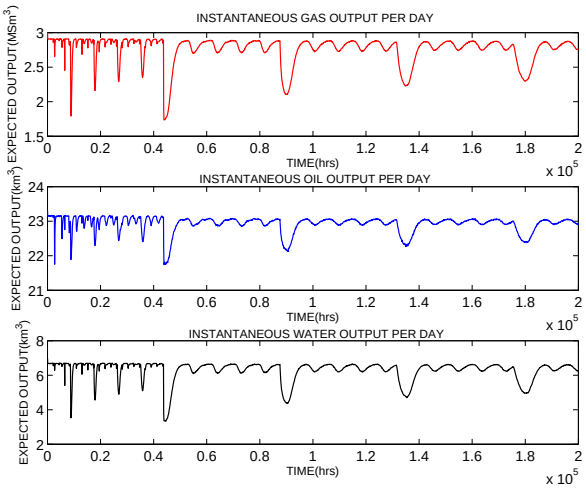


Figure 8: Instantaneous plant performance under CM and PM

unavailability without improving their reliability (Zio, Baraldi, & Patelli 2006). Consequently, the smooth curves in Figure 7 are replaced by ridge curves in Figure 8, the deep drops in performance being due to PM of critical components (e.g., TEG).

5 COMMENTS AND DISCUSSIONS

The simulation produced the same production levels identified via hand calculation by the original authors (Zio, Baraldi, & Patelli 2006) and yielded availability values similar to the reported results. Using 19 cores on a 1895.257MHz AMD Opteron(tm) 6168 processor, cases 1 and 2 took an average of 10.69 minutes and case 3, a few hours.

Though the proposed approach is computationally more demanding than Zio et al's (Zio, Baraldi, & Patelli 2006), it does not require the manual identification of production levels and enumeration of system cut sets. Defining inter-component relationships, component properties and system structure are all it requires, the rest is carried out by efficient algorithms. These attributes, coupled with the fact that it allows system structure to be defined by an adjacency matrix, make it easily applicable to any system structure. However, storage problems are encountered with large systems, since it requires storing, prior to simulation, node flows for all possible system configurations. This makes flow calculation during simulation an inevitable alternative, resulting in increased computational burden. The increased burden, however, can be mitigated by access to parallel computing.

6 CONCLUSIONS

In this paper, an efficient simulation approach to multi-state system performance evaluation has been presented. It's specifically enhanced to model complex architectures, component interdependencies and limited maintenance teams. Its applicability has been demonstrated by analysing a multi-commodity offshore plant originally presented by (Zio, Baraldi, & Patelli 2006). By only defining intra and inter component relationships, the approach provided a similar outcome (within an acceptable time frame) without prior knowledge of the plant's production levels or cut sets. Traditional approaches, however, would require matching each plant configuration to a performance level. This process involves considerable human effort and a detailed knowledge of the plant's operational dynamics. It also suffers the set back of not being sufficiently general and intuitive, as a system's cut-sets and performance levels depend on its structure and the properties of its components. Every system, therefore, would require a unique approach and a unique degree of difficulty. Hence, the proposed approach is an efficient, credible and more intuitive alternative.

ACKNOWLEDGMENTS

The authors would like to acknowledge the gracious support of this work through the EPSRC and ESRC Centre for Doctoral Training on Quantification and Management of Risk & Uncertainty in Complex Systems & Environments.

REFERENCES

Buldyrev, S. V., R. Parshani, G. Paul, H. E. Stanley, & S. Havlin (2010, April). Catastrophic cascade of failures in interdependent networks. *Nature 464*(7291), 1025–1028.

George-Williams, H. & E. Patelli (2016). A hybrid load flow and event driven simulation approach to multi-state system reliability evaluation. *Reliability Engineering & System Safety 152*, 351 – 367.

Levitin, G. (2005). *The Universal Generating Function in Reliability Analysis and Optimization*. Springer-Verlag London Limited.

Mehrotra, S. (1992). On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization 2*(4), 575–601.

Ramirez-Marquez, J. E. & D. W. Coit (2005). A monte-carlo simulation approach for approximating multi-state two-terminal reliability. *Reliability Engineering & System Safety 87*(2), 253 – 264.

Rosato, V., L. Issacharoff, F. Tiriticco, S. Meloni, S. D. Porcellinis, & R. Setola (2008). Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures 4*(1/2), 63+.

Yeh, W.-C. (2015, Dec). An improved sum-of-disjoint-products technique for symbolic multi-state flow network reliability. *Reliability, IEEE Transactions on 64*(4), 1185–1193.

Zang, X., D. Wang, H. Sun, & K. Trivedi (2003, Dec). A bdd-based algorithm for analysis of multistate systems with multistate components. *Computers, IEEE Transactions on 52*(12), 1608–1618.

Zio, E., P. Baraldi, & E. Patelli (2006). Assessment of the availability of an offshore installation by monte carlo simulation. *International Journal of Pressure Vessels and Piping 83*(4), 312 – 320.

Zio, E. & G. Sansavini (2011, March). Modeling interdependent network systems for identifying cascade-safe operating margins. *IEEE Transactions on Reliability 60*(1), 94–101.